Part 6

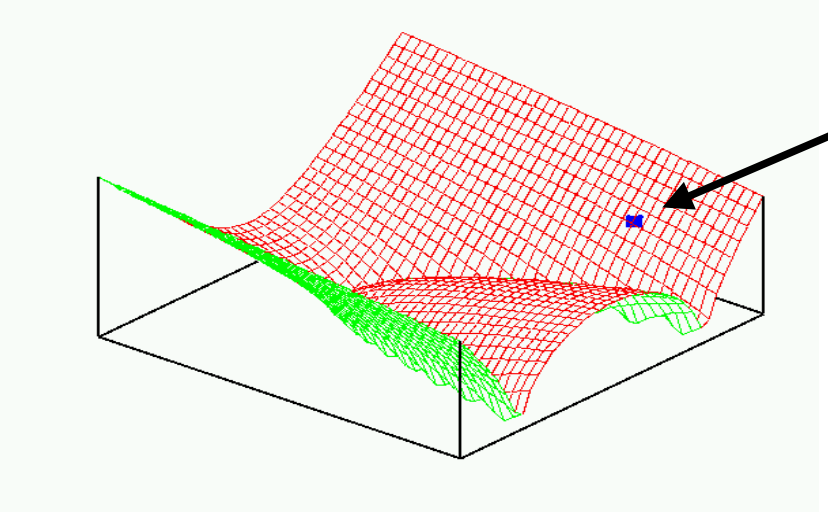
Practical aspects of Newton methods

$$\text{minimize} \quad f(x)$$

What if the Hessian is not positive definite

At the solution, Hessian $\nabla^2 f(x^*)$ is positive definite. If *f(x)* is smooth, Hessian is positive definite near the optimum.

However, this needs not be so far away from the optimum:



At initial point x_0
the Hessian is indefinite:

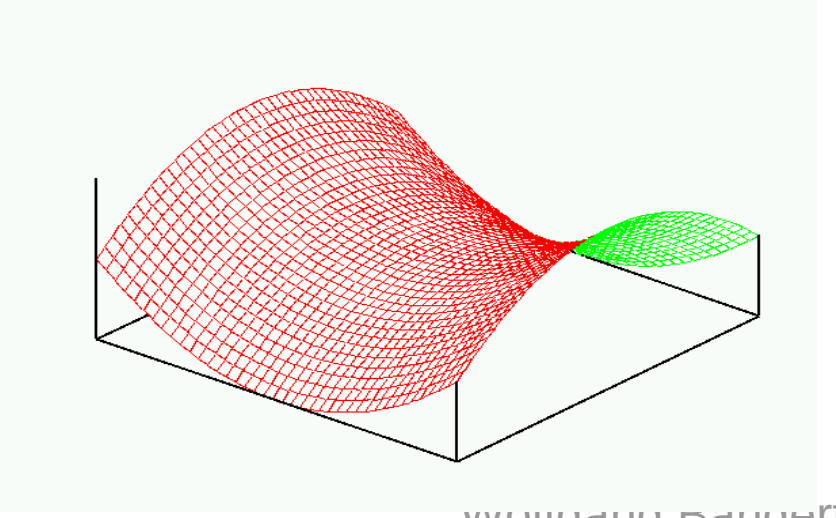
$$H_0 = \nabla^2 f(x_0) = \begin{pmatrix} -0.022 & 0.134 \\ 0.134 & -0.337 \end{pmatrix}$$

$$\lambda_1 = -0.386, \quad \lambda_2 = 0.027$$

Quadratic model

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T H_k p$$

has saddle point instead of minimum, Newton step is invalid!



What if the Hessian is not positive definite

Background: Search direction only useful if it is a descent direction:

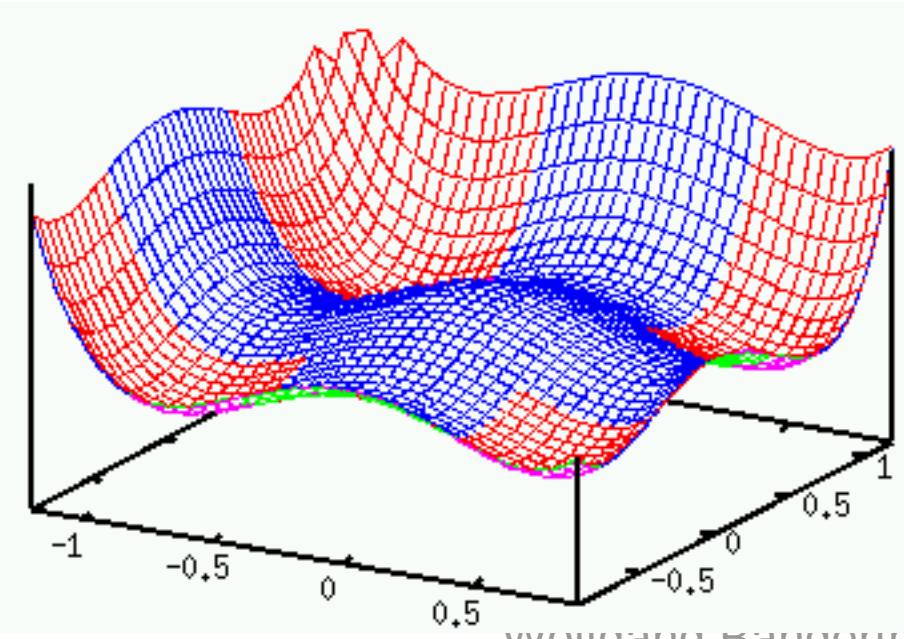
$$\nabla f(x_k)^T \cdot p_k < 0$$

Trivially satisfied for Gradient method, for Newton's method there holds:

$$p_k = -H_k^{-1} g_k \quad \rightarrow \quad g_k^T \cdot p_k = -g_k^T H_k^{-1} g_k < 0$$

Search direction only a guaranteed descent direction, if *H* positive definite!

Otherwise search direction is direction to saddle point of quadratic model and *might* be a direction of *ascent*!



What if the Hessian is not positive definite

If Hessian is not positive definite, then modify the quadratic model:

- retain as much information as possible;
- model should be convex, so that we can seek a minimum.

The general strategy then is to replace the quadratic model by a positive definite one:

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T \tilde{H}_k p$$

Here, $\tilde{H}_k$ is a suitable modification of exact Hessian $H_k = \nabla^2 f(x_k)$ so that $\tilde{H}_k$ is positive definite.

Note: To retain ultimate quadratic convergence, we need that

$$\tilde{H}_k \to H_k \qquad \text{as} \qquad x_k \to x^*$$

What if the Hessian is not positive definite

The **Levenberg-Marquardt** modification:

Choose

$$\tilde{H}_k = H_k + \tau I \qquad \tau > -\lambda_i$$

so that the minimum of

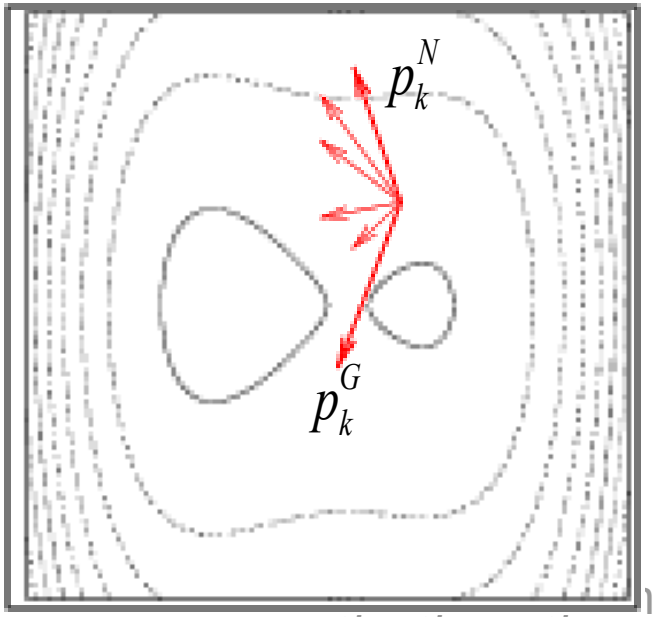
$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T \tilde{H}_k p$$

lies at

$$p_k = -\tilde{H}_k^{-1} g_k = -(H_k + \tau I)^{-1} g_k$$

Note: Search direction is mixture between Newton direction and gradient.

Note: Close to the solution the Hessian must become positive definite and we can choose $\tau = 0$



What if the Hessian is not positive definite

The eigenvalue modification strategy:

Since *H* is symmetric, it has a complete set of eigenvectors:

$$H_k = \nabla^2 f(x_k) = \sum_i \lambda_i v_i v_i^T$$

Therefore replace the quadratic model by a positive definite one:

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T \tilde{H}_k p$$

with

$$\tilde{H}_k = \sum_i \max\{\lambda_i, \epsilon\} \; v_i v_i^T$$

Note: Only modify the Hessian in directions of negative curvature.

Note: Close to the solution, all eigenvalues become positive and we get again the original Newton matrix.

What if the Hessian is not positive definite

One problem with the modification

$$\tilde{H}_k = \sum_i \max\left\{\lambda_i, \epsilon\right\} \; v_i v_i^T$$

is that the search direction is given by

$$p_k = -\tilde{H}_k^{-1} g_k = -\sum_i \frac{1}{\max\left\{\lambda_i, \epsilon\right\}} \; v_i \left(v_i^T g_k\right)$$

that is search direction has *large* component (of size 1/ε) in direction of modified curvatures!

An alternative that avoids this is to use

$$\tilde{H}_k = \sum_i |\lambda_i| v_i v_i^T$$

What if the Hessian is not positive definite

Theorem: Using full step length and either of the Hessian modifications

$$\tilde{H}_k = H_k + \tau I \qquad \tau > -\lambda_i$$

$$\tilde{H}_k = \sum_i \max\left\{\lambda_i, \epsilon\right\} v_i v_i^T$$

we have that if $x_k \to x^*$ and if $f \in C^{2,1}$ then convergence happens with quadratic rate.

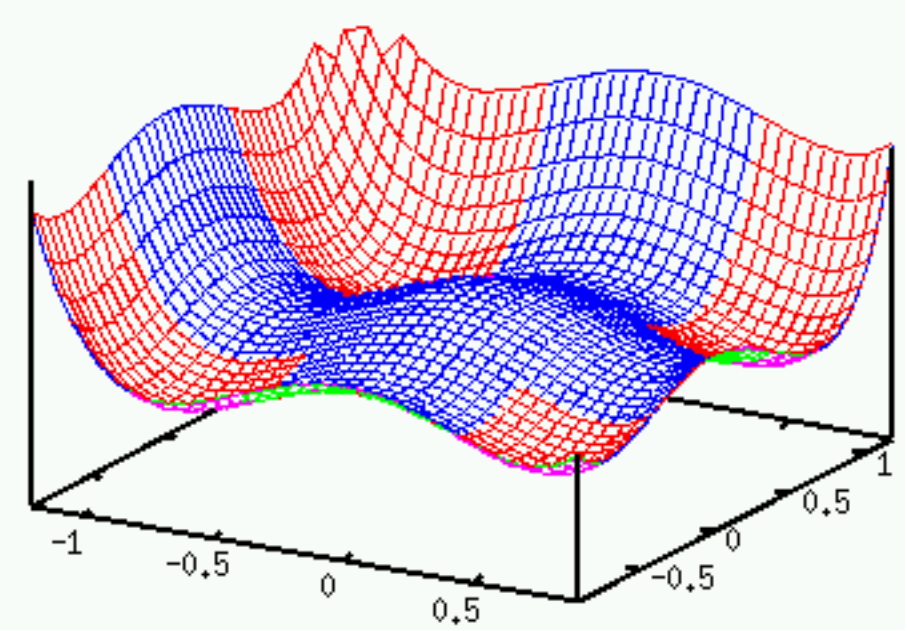
Proof: Since f is twice continuously differentiable, there is a k such that x_k is close enough to x^* that H_k is positive definite.

When that is the case, then

$$\tilde{H}_k = H_k$$

for all following iterations, providing the quadratic convergence rate of the full step Newton method.

What if the Hessian is not positive definite



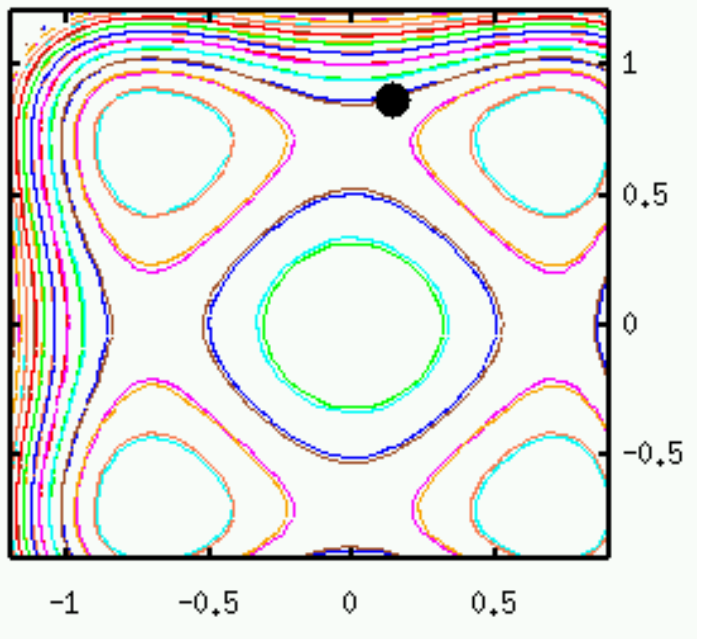
Example:

$$f(x,y) = x^4 - x^2 + y^4 - y^2$$

Blue regions indicate that Hessian

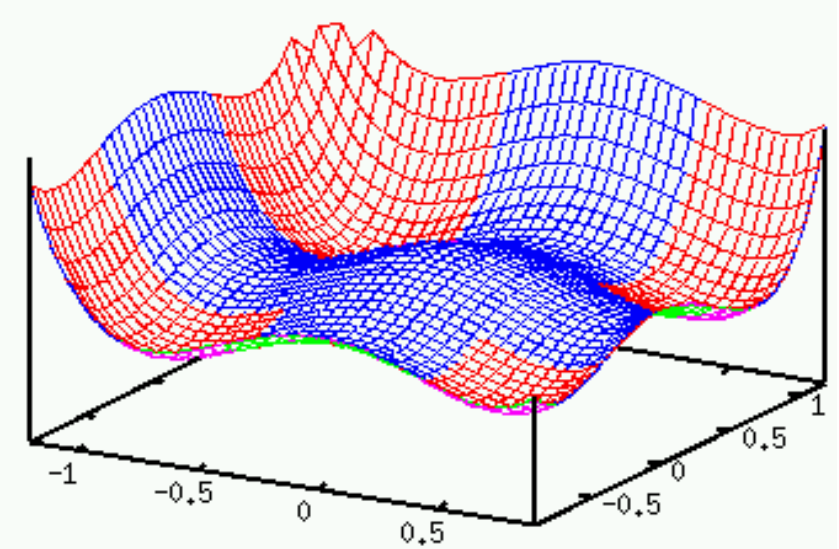
$$\nabla^2 f(x,y) = \begin{pmatrix} 12x^2 - 2 & 0 \\ 0 & 12y^2 - 2 \end{pmatrix}$$

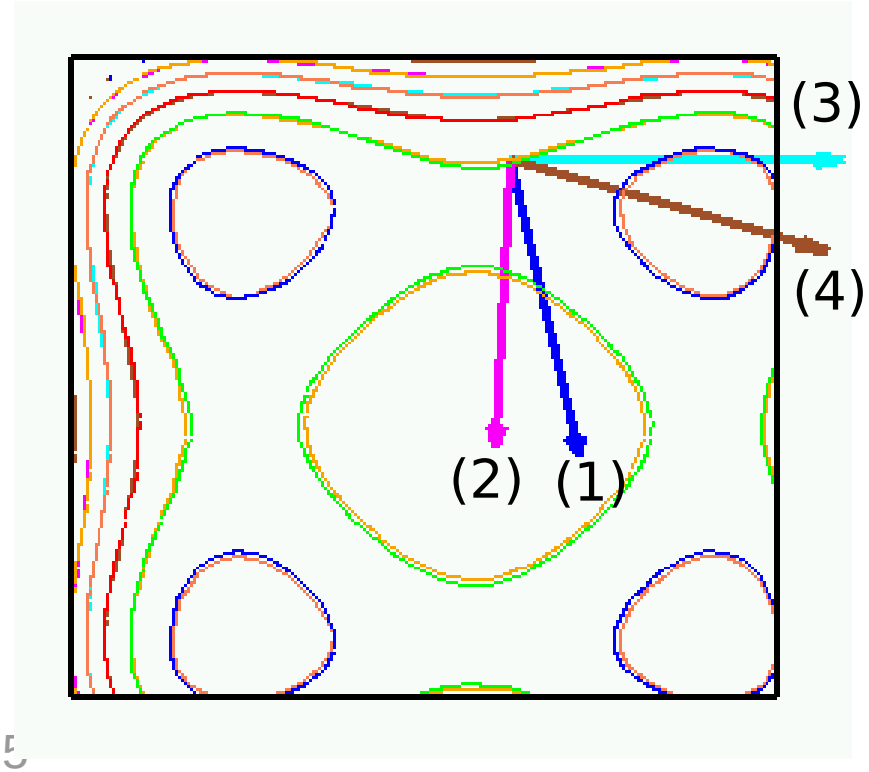
is not positive definite.



minima at $x = \frac{\pm\sqrt{2}}{2,}\; y = \frac{\pm\sqrt{(2)}}{2}$

What if the Hessian is not positive definite





Starting point:

$$x_0 = 0.1 \qquad y_0 = 0.87$$

$$H_0 = \begin{pmatrix} -1.88 & 0 \\ 0 & 7.08 \end{pmatrix}$$

1. Negative gradient
2. Unmodified Hessian search direction
3. Search direction with eigenvalue modified Hessian ($\varepsilon = 10^{-6}$)
4. Search direction with shifted Hessian ($\tau = 2.5$; search direction only good by lucky choice of τ)

Truncated Newton methods

In any Newton or Trust Region method, we have to solve an equation of the sort

$$H_k p_k = -g_k$$

or potentially with a modified Hessian:

$$\tilde{H}_k p_k = -g_k$$

Oftentimes, computing the Hessian is more expensive than inverting it, but not always.

Question: Could we possibly get away with only approximately solving this problem, i.e. finding

$$p_k \approx -H_k^{-1} g_k$$

with suitable conditions on how accurate the approximation is?

Truncated Newton methods

Example: Since the Hessian (or a modified version) is a positive definite matrix, we may want to solve

$$H_k p_k = -g_k$$

using an iterative method such as the Conjugate Gradient method, Gauss-Seidel, Richardson iteration, SSOR, etc etc.

While all these methods eventually converge to the exact Newton direction, we may want to *truncate* this iteration at one point.

Question: When can we terminate this iteration?

Truncated Newton methods

Theorem 1: Let $\hat{p}_k$ be an approximation to the Newton direction defined by

$$H_k p_k = -g_k$$

and let there be a sequence of numbers $\{\eta_k\}, \eta_k < 1$ so that

$$\frac{\|g_k + H_k \hat{p}_k\|}{\|g_k\|} \leq \eta_k < 1$$

Then if $x_k \rightarrow x^*$ then the full step Newton method converges with linear order.

Truncated Newton methods

Theorem 2: Let $\hat{p}_k$ be an approximation to the Newton direction defined by

$$H_k p_k = -g_k$$

and let there be a sequence of numbers $\{\eta_k\}, \eta_k < 1, \eta_k \to 0$ so that

$$\frac{\|g_k + H_k \hat{p}_k\|}{\|g_k\|} \leq \eta_k < 1$$

Then if $x_k \to x^*$ then the full step Newton method converges with superlinear order.

Truncated Newton methods

Theorem 3: Let $\hat{p}_k$ be an approximation to the Newton direction defined by

$$H_k p_k = -g_k$$

and let there be a sequence of numbers $\{\eta_k\}, \eta_k < 1, \eta_k = O(\|g_k\|)$ so that

$$\frac{\|g_k + H_k \hat{p}_k\|}{\|g_k\|} \leq \eta_k < 1$$

Then if $x_k \to x^*$ then the full step Newton method converges with quadratic order.

Part 7

Quasi-Newton update formulas

$$B_{k+1} = B_k + \ldots$$

Quasi-Newton update formulas

Observation 1:

Computing the exact Hessian to determine the Newton search direction

$$H_k p_k = -g_k$$

is expensive, and sometimes impossible.

It *at least* doubles the effort per iteration because we need not only the first but also the second derivative of *f(x)*.

It also requires us to solve a linear system for the search direction.

Quasi-Newton update formulas

Observation 2:

We know that we can get superlinear convergence if we choose the update p_k using

$$B_k p_k = -g_k$$

instead of

$$H_k p_k = -g_k$$

under certain conditions on the matrix B_k.

Quasi-Newton update formulas

Question:

Maybe it is possible to find matrices B_k for which:

- Computing B_k is cheap and requires no additional function evaluations
- Solving

$$B_k p_k = -g_k$$

  for p_k is cheap
- The resulting iteration still converges with superlinear order.

Motivation of ideas

Consider a function *q(x)*.

The **Fundamental Theorem of Calculus** tells us that

$$q(z)-q(x)=\nabla q(\xi)^T\ (z-x)$$

for some $\xi=x+t(z-x),\ \ t\in[0,1]$

Let's apply this to $q(x)=\nabla f(x),\ \ z=x_k,\ \ x=x_{k-1}$:

$$\nabla f(x_k)-\nabla f(x_{k-1})=g_k-g_{k-1}=\nabla^2 f(x_k-t\,\alpha\,p_k)(x_k-x_{k-1})$$
$$=\tilde{H}(x_k-x_{k-1})$$

Let us denote $y_{k-1}=g_k-g_{k-1},\ \ s_{k-1}=x_k-x_{k-1}$ then computing the search direction reads

$$\tilde{H}\,s_{k-1}=y_{k-1}$$

with $\tilde{H}$ the Hessian at some (unknown) intermediate point.

Motivation of ideas

Let us denote

$$y_{k-1} = g_k - g_{k-1} \quad \text{(difference in gradients)}$$
$$s_{k-1} = x_k - x_{k-1} \quad \text{(search direction)}$$

Then computing the search direction reads

$$\tilde{H}\, s_{k-1} = y_{k-1}$$

Goal 1: We don't know what $\tilde{H}$ is (it is the Hessian at some intermediate point). But we know *s* and *y*. Find a way to *estimate* $\tilde{H}$.

Goal 2: Use this approximation to cheaply compute the next search direction!

Motivation of ideas

Requirements:

- We seek a matrix B_{k+1} so that
- The “secant condition” holds:

$$B_{k+1} s_k = y_k$$

- B_{k+1} is symmetric
- B_{k+1} is positive definite
- B_{k+1} changes minimally from B_k
- The update equation is easy to solve for

$$p_{k+1} = -B_{k+1}^{-1} g_{k+1}$$

Davidon-Fletcher-Powell

The DFP update formula:

Given B_k define B_{k+1} by

$$B_{k+1} = \left(I - \gamma\, y_k s_k^T\right) B_k \left(I - \gamma\, s_k y_k^T\right) + \gamma\, y_k y_k^T$$

$$\gamma_k = \frac{1}{y_k^T s_k}$$

This satisfies the conditions:

- It is symmetric and positive definite
- It is among all possible matrices the one that minimizes

$$\left\| \tilde{H}^{-1/2} \left(B_{k+1} - B_k\right) \tilde{H}^{-1/2} \right\|_F$$

- It satisfies the secant condition $B_{k+1} s_k = y_k$

Broyden-Fletcher-Goldfarb-Shanno

The BFGS update formula:

Given B_k define B_{k+1} by

$$B_{k+1} = B_k - \frac{B_k s_k s_K^T B_k}{s_k^T B_k s_K} + \frac{y_k y_k^T}{y_k^T s_k}$$

This satisfies the conditions:

- It is symmetric and positive definite
- It is among all possible matrices the one that minimizes

$$\| \tilde{H}^{1/2} (B_{k+1}^{-1} - B_k^{-1}) \tilde{H}^{1/2} \|_F$$

- It satisfies the secant condition $B_{k+1} s_k = y_k$

Broyden-Fletcher-Goldfarb-Shanno

So far:

- We seek a matrix B_{k+1} so that
- The secant condition holds:

$$B_{k+1} s_k = y_k$$

- B_{k+1} is symmetric
- B_{k+1} is positive definite
- B_{k+1} changes minimally from B_k in some sense
- The update equation is easy to solve for

$$p_k = -B_k^{-1} g_k$$

DFP and BFGS

Now a miracle happens:

For the DFP formula:

$$B_{k+1} = (I - \gamma_k y_k s_k^T) B_k (I - \gamma_k s_k y_k^T) + \gamma_k y_k y_k^T, \qquad \gamma_k = \frac{1}{y_k^T s_k}$$

$$B_{k+1}^{-1} = B_k^{-1} - \frac{B_k^{-1} y_k y_k^T B_k^{-1}}{y_k^T B_k^{-1} y_k} + \frac{s_k s_k^T}{y_k^T s_k}$$

For the BFGS formula:

$$B_{k+1} = B_k - \frac{B_k s_k s_K^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

$$B_{k+1}^{-1} = (I - \rho_k s_k y_k^T) B_k^{-1} (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T, \qquad \rho_k = \frac{1}{y_k^T s_k}$$

This makes computing the next update very cheap!

DFP + BFGS = Broyden class

What if we mixed:

$$B_{k+1}^{DFP} = (I - \gamma_k y_k s_k^T) B_k (I - \gamma_k s_k y_k^T) + \gamma_k y_k y_k^T, \qquad \gamma_k = \frac{1}{y_k^T s_k}$$

$$B_{k+1}^{BFGS} = B_k - \frac{B_k s_k s_K^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

$$B_{k+1} = \phi_k B_{k+1}^{DFP} + (1 - \phi_k) B_k^{BFGS}$$

This is called the “Broyden class” of update formulas.

The class of Broyden methods with $0 \le \phi_k \le 1$ is called the “restricted Broyden class”.

DFP + BFGS = Broyden class

Theorem: Let $f \in C^2$, let x_0 be a starting point so that the set

$$\Omega = \{ x : f(x) \leq f(x_0) \}$$

is convex. Let B_0 be any symmetric positive definite matrix. Then

$$x_k \to x^*$$

for any sequence x_k generated by a quasi-Newton method that uses a Hessian update formula by any member of the restricted Broyden class with the exception of the DFP method $(\phi_k = 1)$.

DFP + BFGS = Broyden class

Theorem: Let $f \in C^{2,1}$. Assume the BFGS updates converge, then

$$x_k \to x^*$$

with superlinear order.

Practical BFGS: Starting matrix

Question: How do we choose the initial matrix B_0 or B_0^{-1} ?

Observation 1: The theorem stated that we will eventually converge for any symmetric, positive definite starting matrix.

In particular, we could choose a multiple of the identity matrix

$$B_0 = \beta I, \quad B_0^{-1} = \frac{1}{\beta} I$$

Observation 2: If β is too small, then

$$p_0 = -B_0^{-1} g_0 = -\frac{1}{\beta} g_0$$

is too large, and we need many trials in line search to find a suitable step length.

Observation 3: The matrices *B* should approximate the Hessian matrix, so they at least need to have the same physical units.

Practical BFGS: Starting matrix

Practical approaches:

Strategy 1: Compute the first gradient g_0, choose a "typical" step length δ, then set

$$B_0 = \frac{\|g_0\|}{\delta} I, \quad B_0^{-1} = \frac{\delta}{\|g_0\|} I$$

so that we get

$$p_0 = -B_0^{-1} g_0 = -\delta \frac{g_0}{\|g_0\|}$$

Strategy 2: Approximate the true Hessian somehow. For example, do one step with the heuristic above, choose

$$B_0 = \frac{y_1^T y_1}{y_1^T s_1} I, \quad B_0^{-1} = \frac{y_1^T s_1}{y_1^T y_1} I$$

and start over again.

Practical BFGS: Limited Memory BFGS (LM-BFGS)

Observation: The matrices

$$B_{k+1} = B_k - \frac{B_k s_k s_K^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

$$B_{k+1}^{-1} = (I - \rho_k s_k y_k^T) B_k^{-1} (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T, \qquad \rho_k = \frac{1}{y_k^T s_k}$$

are full, even if the true Hessian is sparse.

Consequence:

We need to compute all n^2 entries, and store them.

Practical BFGS: Limited Memory BFGS (LM-BFGS)

Solution: Note that in the *k*th iteration, we can write

$$B_k^{-1} = V_{k-1}^T B_{k-1}^{-1} V_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^T$$

$$\text{with } \rho_{k-1} = \frac{1}{y_{k-1}^T s_{k-1}}, V_{k-1} = (I - \rho_{k-1} y_{k-1} s_{k-1}^T)$$

We can expand this recursively:

$$\begin{aligned} B_k^{-1} &= V_{k-1}^T B_{k-1}^{-1} V_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^T \\ &= V_{k-1}^T V_{k-2}^T B_{k-2}^{-1} V_{k-2} V_{k-1} \\ &\quad + \rho_{k-2} V_{k-1}^T s_{k-1} s_{k-2}^T V_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^T \\ &= \ldots \\ &= \left[V_{k-1}^T \cdots V_1^T \right] B_0^{-1} \left[V_1 \cdots V_{k-1} \right] \\ &\quad + \sum_{j=1}^{k} \rho_{k-j} \left\{ \left[V_{k-1}^T \cdots V_{k-j+1}^T \right] s_{k-j} s_{k-j}^T \left[V_{k-j+1} \cdots V_{k-1} \right] \right\} \end{aligned}$$

Consequence: We need only store *kn* entries.

Practical BFGS: Limited Memory BFGS (LM-BFGS)

Problem: *kn* elements may still be quite a lot if we need many iterations. Forming the product with this matrix will then also be expensive.

Solution: Limit memory and CPU time by only storing the last *m* updates:

$$B_k^{-1} = \left[V_{k-1}^T \cdots V_{k-m}^T \right] B_{0,k}^{-1} \left[V_{k-m} \cdots V_{k-1} \right] + \sum_{j=1}^{m} \rho_{k-j} \left\{ \left[V_{k-1}^T \cdots V_{k-j+1}^T \right] s_{k-j} s_{k-j}^T \left[V_{k-j+1} \cdots V_{k-1} \right] \right\}$$

Consequence: We need only store *mn* entries and multiplication with this matrix requires $2mn+O(m^3)$ operations.

Practical BFGS: Limited Memory BFGS (LM-BFGS)

$$B_k^{-1} = \left[V_{k-1}^T \cdots V_{k-m}^T \right] B_{0,k}^{-1} \left[V_{k-m} \cdots V_{k-1} \right] + \sum_{j=1}^{m} \rho_{k-j} \left\{ \left[V_{k-1}^T \cdots V_{k-j+1}^T \right] s_{k-j} s_{k-j}^T \left[V_{k-j+1} \cdots V_{k-1} \right] \right\}$$

In practice:

- Initial matrix can be chosen independently in each iteration; typical approach is again

$$B_{0,k}^{-1} = \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T y_{k-1}} I$$

- Typical values for *m* are between 3 and 30.

Parts 1-7

Summary of methods for smooth unconstrained problems

$$\text{minimize} \quad f(x)$$

Summary

- **Newton's method is unbeatable** with regard to speed of convergence
- **However:** To converge, one needs
  - a line search method + conditions like the Wolfe conditions
  - Hessian matrix modification if it is not positive definite
- Newton's method can be expensive or infeasible if
  - computing Hessians is complicated
  - the number of variables is large
- Quasi-Newton methods, e.g. LM-BFGS, help:
  - only need first derivatives
  - need little memory and no explicit matrix inversions
  - but converge slower (at best superlinear)
- Trust region methods are an alternative to Newton's method but share the same drawbacks